Privacy Preserving Biometric Authentication For Master Password Generation

Proposal for Undergraduate Thesis

Jerry Huang

Abstract

The main way for users to authenticate via biometrics is through 2-factor authentication (2FA), which design requires users to bind their biometric data to a certain device. To solve this problem, Uzun et al. (2021) (to appear in AsiaCCS 2021) proposes a new pipeline which allows for users to enroll their biometric data on any device through a password manager system protocol that requires microphone/camera input. Furthermore, the biometric data stored is privacy preserving, meaning that given the stored password, the attacker unable to recreate any information about the biometric data. The proof of privacy preserving was also done by Uzun et al. (2021). Additionally, Uzun et al. (2021) show that this model (Justitia) is robust against deep learning brute force attacks from randomly sampled faces. Finally, those researchers compare their password management system to a baseline standard, showing that our pipeline has clear benefits and downsides in comparison to password replacing schemes. The research in this paper will focus on the UI system implemented to improve the usability of the existing system developed by Uzun et al. (2021).

1 Introduction

Biometric identification is using the biometric information to authenticate users into a system. For instance, Apple's FaceID or TouchID would fall under authentication via biometric identification. As biometric identification becomes ubiquitous, having one biometrically derived password is a logical continuation to the evolution of biometric authentication. A system such as Apple's FaceID or TouchID requires a Trusted Platform Module (TPM), and cannot share biometric data across hardware. Therefore for a third party to authenticate users using Apple's authentication, 2 factor authentication (2FA) is needed. A solution that allows for 3rd party databases to store biometric data while being privacy preserving (unable to recreate biometric information about the user) is the next step to the 2FA solution.

Currently a typical password manager generates/queries a master password/passphrase, which in turn generates throwaway passwords for each website login. In secure password managers, the throwaway password generating function is privacy preserving, meaning no data of the user's master password, other throwaway passwords, or the user information is leaked. However, a problem arises with this master password, as losing it makes all your account unrecoverable. By having the master password be derived from biometric data, this solves the problem of unrecoverable master passwords. Furthermore, the 2FA solution that requires having a phone is not needed. By presenting a privacy preserving face authentication algorithm to generate master passwords, an authentication method can be developed.

We start with an overview of how to encode biometrics to cryptographic primitives. Cryptographic primitives are standard algorithms conventional to the field of cryptography, such as one-way hash functions. Since biometrics are inherently noisy (i.e. the raw RGB values for a single face taken from two different angles vary considerably), we cannot simply use simple cryptography algorithms to achieve uniformly distributed reproducible keys (Dodis et al., 2004). Uniformly distributed means that every key is equally likely to be the key of the face. More formally, say there is a password system that stores y, and requires a w such that $f(w) = y$ to gain access. Now assume that attackers have access to y, meaning our goal is to design a f such that it is hard to invert $f^{-1}(y)$ to expose w. This the one-way function mentioned previously.

The reason why there are issues using an assumed one-way function is it assumes w is uniformly random. Clearly RGB values of faces are not uniformly random from 0 to 255, and therefore one-way functions has no guarantee that is would be secure on biometric data. Additionally, one-way functions requires an exact w, unsuitable for one-way functions. There are multiple ways to combat this, including key binding, key generation, noninveritible transformations, and salting (Jain et al., 2008). While all these approaches work, the most well-studied is key generation techniques, namely fuzzy extractors. Since we are attempting to generate a uniformly distributed reproducible keys, we use a secure sketch-fuzzy extractor. Formally, a fuzzy extractor is defined as pair of functions, "generate" (Gen) and "reproduce" (Rep) (Dodis et al., 2004). The generate function generates from a biometric input a uniformly random, noise tolerant, string R and helper string P to recover R if lost (Dodis et al., 2004). The reproduce function takes in P and the biometric input and reproduces the same R (Dodis et al., 2004).

While plenty of research has been done on fuzzy extractors, with new advances in computer vision and face recognition, new updates must be made in order to stop attacks on spoofing the biometric input. A very basic example of a spoofing attack would be to print out a piece of paper of someone's face, and attempt to use that on a face detection system. The issue of only using fuzzy extractors themselves is that they do not guarantee security of an adversary, only randomness from the entire input space. Possible attack vectors include recreating a 3d model from public photos (Xu et al., 2016).

The research question that Uzun et al. (2021) had is how create a biometric based password system that is secure given the ever increasing complexity of biometric spoofing. Another key component is the usability of such a system. Given this research question, our hypothesis is that we can create a UI feedback to help users verify through Justitia (Uzun et al., 2021). My research will focus on implementation details on certain parts of this pipeline and also demonstrate a password manager system based on the algorithm our lab has created. The larger implication of this research is to replace the password with a process that is more secure. The main contribution this paper shows is how to improve the usability of the existing pipeline (developed by Uzun), by providing UI feedback to help the user to position and angle their face properly while taking samples for this pipeline.

2 Literature Review

This section will focus on providing background on the work Erkam et. al have done, detailing the pipeline that will help the reader understand the entire Justitia pipeline. Justitia starts with converting biometric data into a more reasonable form. There is an entire field of machine learning to do this, as the study of embeddings, or a low-dimensional vector of a high dimensional space is well studied. Notably, the information that we want is facial and voice data, which both can be encoded into embeddings. There are a few reasons why we want to convert the raw data in embeddings. The main reason is that it reduces unnecessary data that might trip up the fuzzy extractor (a tree in the background of the face capture is not needed). This is why we can afford to lose dimensionality while still preserving information that can discern faces. Next, we go in more in depth on the other alternatives to fuzzy extractors. Lastly, we need to introduce other password replacing measures and the general of how these password systems are judged upon.

2.1 Face and Voice Embeddings

There are various ways to generate facial embeddings. Facial embeddings are related to the research question because they are the part of the input into the fuzzy extractor. Embeddings can be thought of as a metric space with a distance metric such as L2. These embeddings are usually represented by a low dimensional space. For instance, FaceNet uses 128 size vectors (Schroff et al., 2015). The process of converting an image to embedding is usually done through deep convolutional networks, with FaceNet receiving an accuracy of 99.63% on Labelled Faces in the Wild (LFW), a standard dataset used for face detection (Schroff et al., 2015). Other face embedding architectures include DeepFace, which has 97.35% accuracy on Labelled Faces in the Wild (LFW) (Taigman et al., 2014). LFW is a popular dataset that contains labelled faces in the wild. For voice embeddings, one ResCNN based architecture, Deep Speaker, can be modified to output a 128 dimension vector (Taigman et al., 2014).

2.2 Liveness Detection

One common issue with using conventional Convolution Neural Networks (CNNs) is that there is an inability to distinguish whether an input is a real capture of a face or the input is an attacker using something like a photo of a face. This is called a Presentation Attack, and this area of research falls under liveness detection. Various techniques are used to combat this type of attack, such as looking at image quality (Yeh and Chang, 2018). However, countermeasures already have surfaced to combat such image analysis (George et al., 2019). Another popular technique uses hardware such as depth or infrared sensors to tackle liveness (Bhowmik et al., 2011). Lastly, liveness challenges such as requiring the user to to complete a Captcha is another method (Uzun et al., 2018). Such liveness detection systems can be used to enhance the security of such an authentication protocol.

2.3 Privacy-Preserving Biometric Matching

As discussed previously, fuzzy extractors can be used to convert biometric data into a strings R, P, where R is close to uniform, and P contains information about the original biometric input. However, there are other techniques that achieve similar results. Where fuzzy extractors are a biometric cryptosystem that does key generation, there also exists earlier methods that do Key bindings (fuzzy vaults and fuzzy commitment) (Jain et al., 2008). Additionally, there are feature transformations such as biohashing and noninvertible transforms (Jin et al., 2004; Soutar et al., 1999). Biohashing is defined informally as using a randomly

generated key (Jin et al. (2004) gives the example of using a USB stick generating token) along with the biometric data as the input to a noninvertible transformation. The drawbacks from such methods is that they require a secret key K which needs to be maintained in order to distort the data to be privacy preserving (Jain et al., 2008). If somehow K is leaked, then there is no more guarantee of security. Fuzzy extractors have the benefit of great cryptographic guarantees without the need of a secret key K, but it is often difficult to generate keys with high enough stability and entropy.

2.4 Other Password Replacing Measures

There exists other potential methods that attempt to replace passwords without the use of biometrics. There is a certain criteria established by Bonneau et al. (2012), which provides a set of metrics to rate password replacement systems. One example of such a system is a one-time usage QR code system where the user authenticates websites using his phone (Liao et al., 2009). Another such example would be OpenID, a trusted server that can authenticate the users identity.

3 Methodology

This methodology consists of how our lab's implementation of the password system works and is implemented (Uzun et al., 2021). Since the goal of Justitia is to create a password using biometric data, there are quite a few steps in creating a secure system. The first step of our solution is to to use an array of quality filters to ensure things such as liveness. These quality filters sift out basic attacks like using a photo as a face for authentication. Justitia's next step is create embeddings that can represent some biometric data. After that, the database design of the privacy preserving layer is used, where the fuzzy extractor is introduced. Lastly, the master password generation is constructed from the privacy preserving layer.

3.1 Quality Filters and Usability UI

This is the main contribution that will be detailed in this paper. Nonetheless, the design of these quality checks were still theorized by Uzun et al. (2021) and therefore majority their work. The notification to the user of pose, brightness, and orientation through an UI is the main novel aspect of this paper. This was implemented as a website with pure HTML and JavaScript. This design consideration was made in order to provide the user with possible enrollment on as many devices as possible, since majority of devices have a browser capable of running JavaScript and HTML. We first start a capture stream of the camera, giving us a continuous stream of faces. Our first step is to ensure that the signal to noise ratio is high enough for the faces we plan on creating an embedding for. To check for this, we have a few prequality filter checks done by the following steps:

1. Detect the face is present. We use faceLandmark68Net from face-api.js to detect whether or not a face is present. After we detect the face, we detect that face orientation is within a certain pitch, yaw and roll axis. Pitch, yaw, and roll axis are used to determine whether a face is a certain orientation from the camera. If the angle is too far off, the subsequent embedding will not identify the right person. Since the conventional method of pose estimation and solvePnP would likely be to slow (especially in pure JS), we derive a value between -1 and 1 that corresponds to to the pitch, yaw and roll axis. These values are generated from the returned 68 landmarks of faceLandmark68Net. Specifically, $roll'$ is ear marks x value difference divided by the detection box width, $pitch'$ is the jaw to mouth y value difference divided by the detection box height, and yaw' is the difference between the eye median and the nose x value divided by the detection box height. The closer these values are to 0 means the more the face is looking directly at the camera. The UI provides this information to the user such that the user can orient his face correctly. Furthermore, we provide the user with a detection score provided by faceLandmark68Net.

2. Checking adequate brightness by summing the RGB values of a 10 by 10 sample and returning the average of these pixel values. IF the brightness value is above 70, then we return that the brightness is good enough.

3. Checking the orientation and acceleration of the device. We decided to use the accelerator data instead of relying on the gyroscope to include more devices. The user can then check whether their device is too shaky to take samples. This will allow to user to reorient their device to better capture their face.

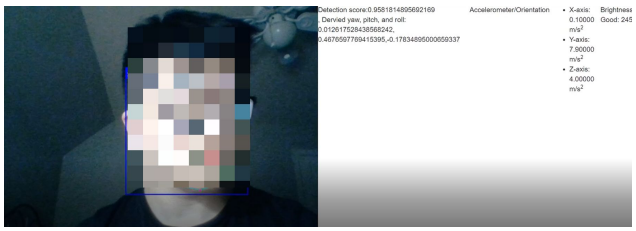

Figure 1: Anonymized example of the UI showing the users derived pitch, yaw, roll, acceleration, and brightness.

These choice were made under consideration that high quality samples should be chosen. The data that passes these initial checks will be sent to an API in which then Justitia will send back a biometric key the user can use for enrollment and authentication. The Justitia API portion of this pipeline was not implemented and is assumed that Erkam et. al will provide such a service for this UI to interact with. This work here is marked as my contribution, and the only claim of work done specifically by me.

3.2 Embeddings

Note that this section was the work of Uzun et al. (2021) and their design. Justitia uses MTCNN and FaceNet to generate facial embeddings. Since MTCNN is one of the benchmarked standards for 2D face alignment in the wild, in addition to be widely cited and used, this choice is obvious (Jin and Tan, 2017). In a survey paper on deep learning based face recognition, FaceNet gets 99.68% accuracy on labelled faces in the wild, while also having the Euclidean distance of two embeddings represent similarity (Guo and Zhang, 2019). This is useful because our voice embedding generator is the same, meaning combining the embeddings is much easier. For voice, Justitia uses DeepSpeaker, a ResCNN based architecture (Li et al., 2017). The output of the combination of these networks is a 128-bit vector, encoding the biometric information of the user's voice or face. The work of implementing and generating the embeddings was done by a larger group of researchers (Uzun et al., 2021).

3.3 Fuzzy Extractor Design

This section was the work of Uzun et al. (2021) and their design. From the previous section, the input to our fuzzy extractor is a 128-bit vector containing biometric data. This biometric data can be compared by euclidean distance. In order for this embedding to be usable by a fuzzy extractor, this data must be converted to hamming distance. To do this, Justitia uses a locality sensitive hash (LSH) approach. LSH is a hashing approach that takes similar values and puts them into the same bucket. Since the embedding vectors are euclidean distance based, this conversion does work. Now since Justitia has a byte vector that uses hamming distance as the new distance metric, Justitia runs this through the fuzzy extractor. The fuzzy extractor allows for vectors who are close in hamming distance to encode into the same secret string. Justitia could simply use this secret key as the master password of our password system, but then the master password would contain biometric information of the user. If this master password would be leaked, then this would not be privacy preserving. Instead, Justitia uses a public-key cryptosystem such as RSA.

Public-key cryptography is defined as having a public key (which can be shared widely) and a private key. Encryption is done with the public key and then only the holder of the private key can decrypt the message. Justitia masks (via XOR) the fuzzy extractor key to be the private key Justitia generated from RSA. Then, Justitia encrypts the masked fuzzy extractor key (aka private key), d, with the public key, storing only the encrypted result, E, and mask used, M, in the database. When the user attempts to authenticate again, the user's biometric input will be first masked by M, then use that as the private key d' to decrypt E. If decrypting E with d' returns d, then the user is authenticated. With this method, the user's biometric data is not compromised even if the master password is leaked. For extra security, this process can be done with multiple biometric vectors, hence the face and voice vectors being two different validation checks. The large part of this algorithm design was done by a larger group of researchers (Uzun et al., 2021).

3.4 Password Management

To go from the above to generating passwords, the KeePass system is used. KeePass is a well known password management system, and many derivatives exists for all devices. Once enrollment of the user is done and the master password pair E, M is generated, the KeePass system then locks the user's passwords with their encryption scheme. This master key can be retrieved by using the biometric information recovery process explained above. After getting the password from the pipeline, the user can use this private key to enroll in their own KeePass system. Authentication then can be done by reusing the web UI to authentication via Justitia's API to retrieve their private key and access their KeePass lock.

4 Results

4.1 Parameter Tuning

This section was the work of Uzun et al. (2021) and their design and experimentation. Since there are a quite a large number of parameters to optimize, it is pretty much impossible to find the absolute best set of parameters. The best set of parameters is defined by the absolute minimum achievable error rate (blocks every malicious attempt). That being said, for the quality filters, each individual parameter such as acceptable brightness can be independently tested. By verifying whether or not which angles and brightness produces acceptable vectors, we found a good pitch, yaw, and roll must be plus or minus 15 degrees relative to the camera. Additionally, they found a brightness value between 90 and 120 to be good, and a statistical score below 47.7 to be acceptable. For my personal experimentation and UI, dummy values were chosen which can later be optimized in a similar fashion. Other parameters such as size of bit vector generated by LSH were also tested, were either an exhaustive search through a defined search space was done or the parameter was set (such as the 128-bit vector dimension). This parameter tuning work was done by a larger group of researchers that were part of this project (Uzun et al., 2021).

4.2 Evaluation

This section was the work of Uzun et al. (2021) and their design and experimentation. Uzun et al. (2021) evaluated on the YouTube Faces (YTF) dataset for facial evaluation (Wolf et al., 2011). By extracting random frames from each video, they see how likely the model will classify correctly. For voice evaluation, they use LibriSpeech (LS) dataset, using 90% of the dataset for evaluation, and 10% for fine-tuning (Panayotov et al., 2015). By looking at the FAR (False Acceptance Rate) and FRR (False Rejection Rate), for each step of the process. FAR is the percentage of users that authenticate without the correct biometrics, while the FRR is the percentage of users that don't authenticate with the correct biometrics. When the quality filters are added, the YTF FRR/FAR percentage decreased 1.68/2.02, respectively. The overall FRR/FAR percents were 0.33/0.00 on YTF and 8.12/0.29 on LS. This evaluation work was done by a larger group of researchers (Uzun et al., 2021).

5 Discussion

This discussion will attempt to show the security guarantees hold true, relating to the research question that the system created is secure in practice. Since this research focuses quite a bit on a practical system, it is important to theorize and attempt use various methods to crack the system. Additionally, it is important to compare other ways that also replace the password and see how this implementation stacks up against other approaches.

5.1 Security Analysis

Note that this section was the work of Uzun et al. (2021) and their design and experimentation. In order to see whether or not the algorithm can deal with fake generated faces (deepfakes), a large corpus of generated faces was made. This was done with creating a pipeline of the filtered YTF faces. Then, StyleGAN was used to create 60 million unique faces. StyleGAN was chosen because of its ability to generate high quality fake faces (Karras et al., 2019). Then, by iterating through all the YTF faces and attempting to authenticate each of the 60 million fake faces, only 2 of the 60 million matched a YTF authenticated face. The probability

here is around random (sampling from a random distribution of faces has the same probability), and shows the strength of the biometric pipeline. This analysis was done by a larger group of researchers that were part of this project.

Although it is possible to develop a more robust attack, without more information on the person, these attacks would be hard to craft for the average person. If the person is a celebrity with large number of samples of their biometric information online, then it could be feasible to construct an attack that successfully manipulates this system. However, this property is not unique to our biometric pipeline, as researchers have cracked many popular biometric scanners using similar methods (Xu et al., 2016). This security analysis work was done by a larger group of researchers (Uzun et al., 2021).

5.2 Comparisons to Other Password Management Methods

Since this research aims to replace a conventional password management system, there needs to be a overall discussion on why this approach has advantages over others. Since the contribution here focuses on usability, it is especially necessary to compare this pipeline's usability to others. Bonneau et al. (2012) has 3 general criteria: usability, deployability, and security. Since this system presented in Sect. 3 can be considered as not a full end-to-end system, this is a theoretical evaluation of what a possible end to end system would look like. In terms of usability, since each user only has to store their own biometric information, we believe that it would have the same usability as other password managers (See: table 1 in Bonneau et al. (2012)). Bonneau et al. (2012) defines each of of the terms below, and given the large (30+ terms) needed to be defined, we refer to the definitions in Bonneau et al. (2012) instead of restating them here.

For deployability, given the combination of password management and biometrics, the pipeline satisfies all but Negligible-Cost-per-User and Maturity. Since the algorithm needs somewhat high quality photos, there is a non negligible cost per user. Maturity is self-explainable, as this system is a research project. Since the KeePass system has been implemented through browser extensions, the Browser-Compatible criteria should be met. For Security, we have the following justifications:

1. Resilient-to-Physical-Observation: Yes, attackers will not be able to extract much biometric samples from physical observations.
2. Resilient-to-Targeted-Impersonation: No, if a person has enough biometric information online, it could be easy to reproduce this.
3. Resilient-to-Throttled-Guessing: Yes, from the security analysis above.
4. Resilient-to-Unthrottled-Guessing: Yes, from the security analysis above.
5. Resilient-to-Internal-Observation: No, the biometric information is vulnerable before storage or during verification. However, the attacker will not be able to extract information if enrollment has already happened.
6. Resilient-to-Leaks-from-Other-Verifiers: Yes, the passwords stored in KeePass are in no relation to the master password.
7. Resilient-to-Phishing: Yes, as password managers themselves are Resilient-to-Phishing since you sign up without your master password.
8. Resilient-to-Theft: Yes, using the same logic as Resilient-to-Phishing.
9. No-Trusted-Third-Party: Yes, users can verify themselves and do not need a 3rd party to verify them.
10. Requiring-Explicit-Consent: Yes, it is impossible to get voice embeddings or turn on the camera without explicit consent from the user.
11. Unlinkable: Yes, the passwords stored in KeePass are in no relation to each other. This means verifiers are unable to determine a person is the same across sites.

In the end, this system provides clear benefit towards previous systems, given that the user keeps somewhat similar biometrics over time and has to correct hardware to use such a system. Over it should be noted that an external 3rd party verifcation protocol like OpenID is another option for the biometric protocol, and solves some of the problems with self-enrollment via a password management design.

6 Conclusion

As many attempts to replace passwords with biometric solutions have been introduced, Uzun et al. (2021) proposed a new solution that is privacy-preserving, and can be inbuilt into a password management system such as KeePass. This system utilizes 4 main parts. The first part, the quality filter is used to reject clear bad samples, using various statistical measures based input images. Then, the embedding generation and LSH is used to transfer the biometric data to something readable by a fuzzy extractor. After the fuzzy extractor, the password generated is used as a master password for a KeePass system. This pipeline is a novel contribution to the field of authentication, with many ideas able to be borrowed from. Notably, the contribution of the usability UI expands on Uzun et al. (2021) in order to make the pipeline more accessible to users.

One of the main concerns of the approach presented by Uzun et al. (2021) is utilizing things such as deepfakes to impersonate a user. Their system is shown to be robust against novel deep learning attacks, though more research would need to be done to combat more targeted attacks on people with large amounts of public biometric data. In terms of implementation, the combination of biometrics with a password manager has clear benefits and downsides, though no system is perfect in all regards. The usability UI expanded on Uzun et al. (2021) can be more published to be presentable to the average user who don't know about pitch, pay, and roll. In conclusion, there are some interesting ideas here that can be used to further the advent of authentication, further expanding on Uzun et al. (2021) ideas, such as the usability UI presented.

7 Acknowledgments

Thanks for Erkam Uzun, Wenke Lee, and Pak Ho (Simon) Chung for their guidance and the authors of the AsiaCCS Justitia paper for the design and implementation of the Justitia pipeline, in which without, this paper would not have been possible. Large majority (99.9%) of this paper is their excellent work.

References

Bhowmik, M. K., K. Saha, S. Majumder, G. Majumder, A. Saha, A. N. Sarma, D. Bhattacharjee, D. K. Basu, and M. Nasipuri (2011). Thermal infrared face recognition—a biometric identification technique for robust security system. *Reviews, refinements and new ideas in face recognition 7*.

Bonneau, J., C. Herley, P. C. Van Oorschot, and F. Stajano (2012). The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE Symposium on Security and Privacy*, pp. 553–567. IEEE.

Dodis, Y., L. Reyzin, and A. Smith (2004). Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *International conference on the theory and applications of cryptographic techniques*, pp. 523–540. Springer.

George, A., Z. Mostaani, D. Geissenbuhler, O. Nikisins, A. Anjos, and S. Marcel (2019). Biometric face presentation attack detection with multi-channel convolutional neural network. *IEEE Transactions on Information Forensics and Security 15*, 42–55.

Guo, G. and N. Zhang (2019). A survey on deep learning based face recognition. *Computer Vision and Image Understanding 189*, 102805.

Jain, A. K., K. Nandakumar, and A. Nagar (2008). Biometric template security. *EURASIP Journal on advances in signal processing 2008*, 1–17.

Jin, A. T. B., D. N. C. Ling, and A. Goh (2004). Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern recognition 37*(11), 2245–2255.

Jin, X. and X. Tan (2017). Face alignment in-the-wild: A survey. *Computer Vision and Image Understanding 162*, 1–22.

Karras, T., S. Laine, and T. Aila (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410.

Li, C., X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu (2017). Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304 650*.

Liao, K.-C., W.-H. Lee, M.-H. Sung, and T.-C. Lin (2009). A one-time password scheme with qr-code based on mobile phone. In *2009 Fifth International Joint Conference on INC, IMS and IDC*, pp. 2069–2071. IEEE.

Panayotov, V., G. Chen, D. Povey, and S. Khudanpur (2015). Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210. IEEE.

Schroff, F., D. Kalenichenko, and J. Philbin (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823.

Soutar, C., D. Roberge, A. Stoianov, R. Gilroy, and B. V. Kumar (1999). Biometric encryption. In *ICSA guide to Cryptography*, Volume 22. McGraw-Hill.

Taigman, Y., M. Yang, M. Ranzato, and L. Wolf (2014). Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708.

Uzun, E., S. P. H. Chung, I. Essa, and W. Lee (2018). rtCaptcha: A real-time CAPTCHA based liveness detection system. In *NDSS*.

Uzun, E., C. Yagemann, S. Chung, V. Kolesnikov, and W. Lee (2021). Cryptographic key derivation from biometric inferences for remote authentication. In *AsiaCCS*.

Wolf, L., T. Hassner, and I. Maoz (2011). Face recognition in unconstrained videos with matched background similarity. In *CVPR 2011*, pp. 529–534. IEEE.

Xu, Y., T. Price, J.-M. Frahm, and F. Monrose (2016). Virtual u: Defeating face liveness detection by building virtual models from your public photos. In *25th {USENIX} Security Symposium (USENIX Security 16)*, pp. 497–512.

Yeh, C.-H. and H.-H. Chang (2018). Face liveness detection based on perceptual image quality assessment features with multi-scale analysis. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 49–56. IEEE.